Algebraic Tailoring of Discontinuous Galerkin p-Multigrid for Convection

Krzysztof Fidkowski*

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109

Abstract

This work presents an element-local algebraic approach to constructing coarse spaces for p-multigrid solvers and preconditioners of high-order discontinuous Galerkin discretizations. The target class of problems is convective systems on unstructured meshes, a class for which traditional p-multigrid typically fails to reach textbook multigrid efficiency due to a mismatch between smoothers and coarse spaces. Smoothers that attempt to alleviate this mismatch, such as line-implicit, incomplete LU, or Gauss-Seidel, deteriorate on grids that are not aligned with the flow, and they rely on sequential operations that do not scale well to distributed-memory architectures. In this work we shift attention from the smoothers to the coarse spaces, and we present an algebraic definition of the coarse spaces within each element based on a singular-value decomposition of the neighbor influence matrix. On each multigrid level, we employ a block-Jacobi smoother, which maintains algorithmic scalability as all elements can be updated in parallel. We demonstrate the performance of our solver on discretizations of advection and the linearized compressible Euler equations.

Keywords: p-Multigrid, Discontinuous Galerkin, Convection, Singular-Value Decomposition

*Corresponding author

Email address: kfid@umich.edu, (tel) 734-615-7247, (fax) 734-763-0578 (Krzysztof Fidkowski)

1. Introduction

1.1. Motivation

Compared to traditional second-order methods, high-order discretizations offer accuracy and efficiency advantages for many problems of engineering interest. These include convection-dominated flows in aerospace engineering, which demand accurate resolution of features such as boundary layers, vortices, and acoustic waves. One popular high-order method is discontinuous Galerkin (DG) [1, 2], and this will be the subject of the present work. DG offers attractive features such as convective stability, support of unstructured meshes, simple parallelization, and a variational formulation for error estimation. Recent DG extensions [3, 4] share these features, and we focus on standard DG chiefly because it has been the subject of a large body of literature on discretizations and solvers.

One drawback of DG is its expense, and largely for this reason DG has not yet become a mainstream method such as second-order finite volume. DG does consume many degrees of freedom (DOFs) per element of a mesh, especially compared to continuous methods which do not include inter-element jumps in the approximation space. However, DOF consumption by itself is not the whole story, since for many problems DG fares well compared to other discretizations in a DOF versus accuracy tradeoff [5]. In addition to DOFs, we need to consider solvers, and this is where DG can become expensive. The stiff discrete systems arising from high-order discretizations call for efficient implicit solvers that balance several considerations, including iteration count, operation count, memory usage, memory access, and scalability. Designing such solvers for DG is not easy because of the relatively high bandwidth of the systems, even when using element-wise compact stencils.

For nonlinear problems, Newton-Krylov is commonly used, and its performance hinges on an effective preconditioner. For high-order DG discretizations, p-multigrid has been investigated as both a preconditioner [6, 7] and as a stand-alone solver [8, 9, 10, 11].

In this work we consider the former application of p-multigrid.

The tenet of p-multigrid is simple: lower order discretizations serve as coarse-spaces in a multigrid correction procedure for a high-order solution. This procedure requires an iterative smoother, and finding smoothers that complement the coarse-order spaces and that scale well in parallel is not easy, especially for multi-dimensional convection-dominated problems on unstructured meshes. Whereas most previous work in p-multigrid has focused on iterative smoothers, in the present work we shift attention to the coarse spaces, which in an algebraic interpretation need not be strict low-order discretizations. We show that by choosing the coarse spaces independently on each element, we can achieve excellent multigrid performance even with simple smoothers such as block Jacobi.

This paper is organized as follows. In the remainder of this section, we present background material on p-multigrid. Section 2 reviews some key properties of p-multigrid combined with typical smoothers. Section 3 presents the proposed algebraic approach to selecting a coarse space inside an element. Finally, Section 4 shows results for advection and the linearized Euler equations.

1.2. Background

In classical multigrid techniques, solutions on spatially coarser grids are used to correct solutions on the fine grid. The use of coarser grids is motivated by the observation that local smoothers generally do a poor job at eliminating low-frequency error modes on the fine grid. However, these low-frequency error modes can be effectively corrected by smoothing on coarser grids, in which they appear as high-frequency. In p-multigrid, the idea is similar, with the exception that lower-order approximations serve as the "coarse grids."

The idea of using low-order discretizations to correct high-order solutions through a multigrid approach was pioneered by the spectral finite element community for elliptic equations. In the early 1980's, Zang *et al* [12, 13] applied a multigrid approach

to Fourier spectral approximations with periodic and Dirichlet boundary conditions and demonstrated improvement over simple iterative smoothers. A more general multigrid approach, applied to the geometrically-flexible spectral finite element method, was introduced in 1987 by Rønquist and Patera [14]. For a one-dimensional Poisson problem, these authors demonstrated convergence rates independent of the number of elements and of the order, for spectral multigrid with simple and parallelizable diagonal Jacobi smoothers. In follow-up papers, Maday and Muñoz [15, 16] justified these convergence rates and proved order-independence in one dimension. However, this order-independence does not extend to multiple dimensions [16], an observation that has been attributed to the deterioration of the diagonal smoother with increasing order, and one that has seen been addressed through under-relaxation, semi-coarsening, and line relaxation [17, 18, 19, 20].

The application of p-multigrid to discontinuous Galerkin finite elements is more recent. p-Multigrid fits naturally within the framework of DG, due to the flexibility in approximation order of these discretizations. Additional coarse grid information is not required since the same spatial grid is used by all levels. In addition, a hierarchical basis can be used, eliminating the duplication of state storage at each level. Although the premise of the method remains similar to the spectral element case, there are key differences, including:

- *The character of the target differential equation*: whereas spectral elements have been primarily applied elliptic equations, the niche for DG lies in convection-dominated flows.

- *The maximum expected approximation order*: unlike the very large orders common in spectral methods, $p = 3$ or $p = 4$ are already considered "high order" by DG practitioners.

- *The availability of smoothers*: with the relatively lower approximation orders common in many DG applications, element Jacobi smoothers, which treat the

unknowns within each element implicitly, are computationally feasible.

These differences are sufficient to have sparked recent interest in p-multigrid methods for DG discretizations. One of the first studies was performed by Helenbrook *et al* [8], who analyzed p-multigrid for DG in one and two dimensions. For a two-level cycle, these authors demonstrated order and mesh independence for scalar convection in one dimension when using element Jacobi smoothing. In two dimensions, they found that order independence persisted, but that convergence depended on the number of elements. Around the same time, the author presented p-multigrid results for the compressible Euler equations of gas dynamics in two dimensions [9, 21]. This work confirmed order independence and mesh dependence for equation systems when using element Jacobi smoothing. It also presented a more powerful element-line Jacobi smoothing strategy, in which unknowns on a contiguous line of elements are treated implicitly, that significantly improved convergence and in some cases yielded nearly mesh-independent V-cycle rates in combination with p-multigrid. The element-line Jacobi solver was further analyzed in two dimensions by Mascarenhas *et al* [22], who confirmed mesh independence and suggested alternating the direction of lines for improved robustness when the flow is not aligned with the mesh. The use of other smoothers, including explicit Runge-Kutta is investigated by Luo *et al* [23].

Nastase and Mavriplis [24] present a p-then-h multigrid strategy in which they employ h-multigrid mesh coarsening to create more multigrid levels beyond the coarsest approximation order. This strategy addresses the expense of solving the coarsest approximation order (e.g. $p = 0$) problem directly or smoothing on it many times. It does not address h-dependence of the solver as a whole, as even with an exact coarse-order solve, h-dependence persists [8, 21].

p-Multigrid has also been investigated for DG discretizations of singularly-perturbed convection-diffusion equations, including the compressible Navier-Stokes system [25, 26, 10, 27], and of pure diffusion equations [28, 11]. Order-independence of p-multigrid

with element-Jacobi smoothing is observed in a majority of these one and two-dimensional studies, and h-dependence appears less prominent.

Finally, aside from a couple small problems considered by the author [21], there has been little research into the application of p-multigrid to three-dimensional problems. At stake is the possible degradation of convergence due to the smoothers becoming less powerful: for example, an element-line Jacobi smoother is an exact solve in one dimension, but not in two or three dimensions.

2. Properties of p-Multigrid

2.1. Overview

The simplest multigrid iteration is a two-level correction scheme, illustrated for a linear problem in Figure 1.

[Figure 1 about here.]

The fine-level system to be solved is

$$\mathbf{A}_h \mathbf{u}_h = \mathbf{f}_h,$$

where $\mathbf{A}_h \in \mathbb{R}^{N_h \times N_h}$ is the system matrix, $\mathbf{u}_h \in \mathbb{R}^{N_h}$ is the state, and $\mathbf{f}_h \in \mathbb{R}^{N_h}$ is the forcing vector. In a two-level correction scheme, we iterate towards the solution by using a combination of a smoother on the fine level and a correction obtained from a coarse-level solution. We discuss options for the iterative smoother in the next subsection. The coarse level is defined by a prolongation operator, $\mathbf{I}_h^H \in \mathbb{R}^{N_h \times N_H}$, the columns of which define each coarse-level state as a linear combination of fine-level states, and possibly by a separate residual restriction operator, $\mathbf{I}_H^h \in \mathbb{R}^{N_H \times N_h}$, the rows of which prescribe how the fine-level equations are linearly combined to form the coarse-level equations. In a Galerkin formulation, $\mathbf{I}_H^h = \left(\mathbf{I}_h^H\right)^T$. The coarse-level operator is then given by

$$\mathbf{A}_H = \mathbf{I}_H^h \mathbf{A}_h \mathbf{I}_h^H$$

The two-level scheme can be summarized in the following steps:

1. Smooth $\mathbf{u}_h$ ν_{pre} times on the fine grid
2. Compute residual $\mathbf{r}_h = \mathbf{f}_h - \mathbf{A}_h\mathbf{u}_h$, and restrict it: $\mathbf{r}_H = \mathbf{I}_H^h\mathbf{r}_h$
3. Solve for the error on the coarse grid, $\mathbf{A}_H\mathbf{e}_H = \mathbf{r}_H$
4. Prolongate $\mathbf{e}_h = \mathbf{I}_h^H\mathbf{e}_H$ and correct $\mathbf{u}_h = \mathbf{u}_h + \mathbf{e}_h$
5. Smooth $\mathbf{u}_h$ ν_{post} times on the fine grid

2.2. Smoothers

The choice of smoother heavily influences the multigrid convergence rate. Popular smoothers for p-multigrid DG include element-block Jacobi/Gauss-Seidel, element-line, alternating-direction implicit (ADI), incomplete lower-upper (ILU) decomposition, and multistage schemes. [8, 21, 10, 11]. These smoothers differentiate themselves in a performance versus cost tradeoff. An additional consideration, however, is algorithmic scalability, and generally speaking, powerful smoothers typically do not scale well.

In this work we focus on two representative smoothers: element-block Jacobi ("block") and element-line Jacobi ("line"). The block smoother is one of the simplest scalable methods, and the line smoother is similar to block Gauss-Seidel and ILU. For the linear system $\mathbf{Au} = \mathbf{f}$, both can be expressed in the following form,

$$\mathbf{u}^{n+1} = \mathbf{u}^n - \mathbf{M}^{-1}\mathbf{r}(\mathbf{u}^n), \qquad \mathbf{r}(\mathbf{u}^n) \equiv \mathbf{A}\mathbf{u}^n - \mathbf{f}$$

where n is the iteration number and $\mathbf{M}$ is the preconditioner that defines the smoother. For the block smoother, $\mathbf{M}$ is the block diagonal of $\mathbf{A}$, and for the line smoother $\mathbf{M}$ consists of block tridiagonal systems formed by stringing elements together, as illustrated in Figure 2.

[Figure 2 about here.]

2.3. Dependence on h

To demonstrate the performance of a two-level p-multigrid scheme for a practical problem, we consider flow governed by the nonlinear Euler equations over a bump geometry, as illustrated in Figure 3.

[Figure 3 about here.]

Full-approximation storage [29] is used for the nonlinear extension of the multigrid algorithm described at the beginning of this section. As the two levels, we consider $p_h = 3$ and $p_H = 2$ approximation orders.

We first consider block smoothing. Figure 4 shows residual convergence of two-level multigrid on a sequence of grid refinements. The initial condition in each case is uniform free-stream.

[Figure 4 about here.]

As shown, the residual converges in two phases, the first of which is a near plateau whose duration depends on the grid size (h). The steepness of the drop in the second phase also depends on the grid size. A large number of smoothing iterations on the fine space can mask this h-dependence, for a limited time – as the grid is refined, h-dependence eventually appears.

To gain insight into the performance deterioration of p-multigrid with block smoothing, we visualize the slowest converging error mode (density component) in Figure 5. This mode is measured on the finest grid at the end of the near-plateau phase – i.e. before the steep drop. As shown, the error is low frequency in the streamwise direction and high-frequency perpendicular to the streamlines. This result suggests that a more effective smoother could be designed by treating the streamwise parallel/perpendicular directions differently.

[Figure 5 about here.]

The line smoother addresses the multigrid h-dependence for this problem by solving for all unknowns in the streamwise direction at once. The resulting residual convergence histories are shown in Figure 6

[Figure 6 about here.]

The great performance of the line solver in this bump case is due to the underlying structured mesh that allows for long element lines aligned with the flow direction. When this is not the case, for example on general unstructured meshes, line smoothing no longer cures h-dependence. Figure 7 illustrates the effect of line/flow misalignment for a linear advection problem on a structured mesh with horizontal lines. As the flow angle, α, increases convergence deteriorates and h-dependence becomes more prominent.

[Figure 7 about here.]

Three-dimensional problems also pose challenges to the line smoother and its variants. Figure 8 shows h-dependence of the two-level p-multigrid cycle with line smoothing for flow over a wing: As the mesh is refined, the convergence rate drops.

[Figure 8 about here.]

2.4. Scalability

The line smoother is algorithmically more powerful than the block smoother because it damps out more modes through its implicit treatment of multiple elements linked together. This property is important for p-multigrid because the more work the smoother does, the less work remains for the coarse-level correction. ILU and block Gauss-Seidel smoothing share similar characteristics. In addition, even though the line smoother is more computationally intensive than the block smoother, the algorithmic performance benefits typically outweigh these costs [21].

However, a problem with line smoothing is that it is not algorithmically scalable because it relies on a non-local coupling of elements. Figure 9 shows how partitioning breaks lines, and shorter lines degrade performance. In the limiting case of single-element lines, the line smoother reduces to the block smoother.

[Figure 9 about here.]

On the other hand, block smoothing scales very well as all elements can be operated on simultaneously and since only nearest-neighbor information is required. Since block smoothing does not damp out as many modes as line smoothing, if we resign ourselves to use block smoothing we need to compensate with more work elsewhere. In the present paper we pursue this idea in the context of p-multigrid by offloading work to the coarse levels. In particular, we equip the coarse levels with the ability to represent error modes not addressed by the smoother. This idea of tuning the coarse levels to complement the smoother is similar to the premise of algebraic multigrid (AMG), although designing efficient and scalable "black-box" AMG solvers for convection problems is challenging. We thus propose a more local, element-centric, approach driven by an analysis of inter-element coupling, as discussed in the next section.

2.5. Semi-coarsening in p

Before presenting the general coarse-level selection procedure, we revisit the bump-flow problem of subsection 2.3. As we saw in Figure 5, when using p-multigrid with the block smoother, the slowest converging mode was high-frequency perpendicular to the streamlines. A somewhat contrived but illustrative solution to this slow convergence is to equip the coarse space with the ability to represent such error modes. To do this, we consider a quadrilateral mesh aligned with the flow, on which we use an anisotropic order approximation for the coarse space: $p = 3$ in the flow-perpendicular direction and $p = 2$ in the streamwise direction, as shown in Figure 10. In contrast, a standard coarse space would be $p = 2$ in both directions.

[Figure 10 about here.]

When using the "semi-coarsened" approximation space as the coarse level, the problematic error modes are effectively eliminated after the coarse-level correction. As a result, h-independent convergence rates are recovered – see Figure 11.

[Figure 11 about here.]

Generalizing order semi-coarsening to unstructured meshes and non-flow-aligned elements is possible but not trivial. Such an algorithm would have to discover the appropriate directions for semi-coarsening and would have to decide on the appropriate orders in each direction. It is also not clear that such principal orthogonal directions always exist, especially in three dimensions. Thus, we seek a more automated and systematic approach to the coarse-level construction, as described in the next section.

3. SVD-Based Coarse Levels

In this section we present a definition of the coarse levels in p-multigrid that better represents modes not addressed by the smoother. For scalability, we employ the block smoother, and we define coarse levels locally for each element. Furthermore, we restrict our attention to a two-level correction scheme.

3.1. Excitable Modes

At each iteration, the block smoother applied to an element makes the solution in that element correct with respect to the "boundary conditions" imposed by surrounding elements. Of course, those boundary conditions change from iteration to iteration as error modes make their way out of the domain and the solution converges globally. Our motivating observation is that not all degrees of freedom (or "modes") inside an element are created equal with respect to this error propagation. For example, in many convective flows, there exist solution modes inside an element that are not affected by state variations on surrounding neighbors. These modes then do not play a role in inter-element error propagation and they can be trimmed from the coarse-level approximation space without sacrificing any approximation power of the coarse level when coupled with the block smoother. Similarly, there will be some modes that are only affected slightly by neighbor perturbations, and we could consider trimming these too. To make this process systematic, we need to identify and rank such modes.

We take a matrix-based approach in identifying solution modes on each element that are important for inter-element propagation of error. Denote by $\mathbf{U}$ the discrete state vector, $\mathbf{R}$ the discrete residual vector, and $\mathbf{A} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ the residual Jacobian matrix. These quantities are associated with the fine-level, but we drop the subscript h for clarity of exposition. Consider an element i, with state $\mathbf{U}_i$ and residual $\mathbf{R}_i$. Figure 12(a) highlights the portion of $\mathbf{A}$ relevant to element i.

[Figure 12 about here.]

The block row of $\mathbf{A}$ associated with the residuals on element i contains a self block $\mathbf{A}_{i,i}$, which dictates how state perturbations on element i affect the residual $\mathbf{R}_i$, as well as off-diagonal blocks $\mathbf{A}_{i,j}$, which dictate how state perturbations on neighbors j affect $\mathbf{R}_i$. There will only be a few j values for which $\mathbf{A}_{i,j}$ is nonzero since typical element shapes have only a small number of neighbors. We can therefore compress the entire block row into a smaller dense matrix, $\mathbf{A}_{i,:}$ as shown in Figure 12(b).

The compressed off-diagonal matrix, $\mathbf{A}_{i,:} \backslash \mathbf{A}_{i,i}$ maps neighbor state perturbations onto residuals in element i. In determining modes in element i that are important for inter-element error transport, we focus on this matrix. Specifically, we ask: which residual modes in element i are most "excitable" by neighbors state perturbations. Once we identify the important residual modes, we can map these locally to state modes using $\mathbf{A}_{ii}^{-1}$. To answer this question, we take a singular value decomposition (SVD) of $\mathbf{A}_{i,:} \backslash \mathbf{A}_{i,i}$, as illustrated in Figure 13.

[Figure 13 about here.]

The SVD provides information on the neighbor modes that most strongly affect residuals in element i. Currently we actually neglect this information on neighbor modes, stored in the matrix $\mathbb{V}_i$, as we are primarily interested in the corresponding residual modes in element i, i.e. the columns of $\mathbb{U}_i$.

The columns of $\mathbb{U}_i$ contain element i residual modes ranked in order of importance with respect to how excitable they are by neighboring state perturbations. The cor-

responding singular values σ_k provide the weights in this ranking. Now, considering element i in isolation, the states that map to the residual modes in $\mathbb{U}_i$ are given by the columns of $\mathbf{\Phi}_i$, where

$$\mathbf{\Phi}_i = \mathbf{A}_{ii}^{-1}\mathbb{U}_i.$$

At a discrete level, the columns of $\mathbf{\Phi}_i$ contain state modes that are most strongly excited by perturbations on the neighbors. When the singular values σ_k are disparate, the most important modes to retain in the coarse levels will be in the first columns of $\mathbf{\Phi}_i$.

A visual demonstration of the results of this singular value decomposition for a 2D advection problem is shown in Figure 14. The triangular element of interest, shown shaded in Figure 14(a), is oriented in a manner such that it is only affected by the state in one neighbor. When using a $p = 4$ DG approximation, the number of unknowns in each triangle is $r = 15$. However, the element of interest is only affected by the state on its single upwind edge, and the number of possible modes on this edge is $p+1 = 5$. As a result, only 5 residual modes on the shaded element are excited by neighbor perturbations. This is confirmed by the SVD calculations, which yields 5 nonzero singular values, as illustrated in the overlaid singular value plot in Figure 14(a). The corresponding state modes, i.e. the first 5 columns of $\mathbf{\Phi}_i$, are illustrated in Figure 14(b).

[Figure 14 about here.]

3.2. Petrov-Galerkin Projection

The singular value decomposition of the neighbor influence matrix gives us a ranking of the most important residual modes to keep on the coarse level from the point of view of preserving inter-element communication. These residual modes are orthonormal vectors in the columns of $\mathbb{U}_i$. To ensure that we test these modes on the coarse levels, we choose the columns of $\mathbb{U}_i$ as the test vectors. That is, the coarse-level equations are linear combinations of the fine-level equations according to coefficients in the columns of $\mathbb{U}_i$.

Performing the SVD for every element, we obtain element-specific test vectors, $\mathbb{U}_i$, and basis functions, $\mathbf{\Phi}_i$. We roll these matrices into global restriction ($\mathbf{R}$) and prolongation ($\mathbf{P}$) operators as,

$$\mathbf{R} = \begin{bmatrix} \mathbb{U}_1^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbb{U}_2^T & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbb{U}_N^T \end{bmatrix}, \qquad \mathbf{P} = \begin{bmatrix} \mathbf{\Phi}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{\Phi}_N \end{bmatrix},$$

where N is the number of elements. We now define a transformed system matrix as

$$\widetilde{\mathbf{A}} = \mathbf{RAP}.$$

Note that $\mathbf{R}$ and $\mathbf{P}$ are square matrices, so that $\widetilde{\mathbf{A}}$ is the same size as $\mathbf{A}$ – this is only a basis transformation, not yet a coarse-space selection. In addition, the block diagonals of $\widetilde{\mathbf{A}}$ are identity matrices, since

$$\widetilde{\mathbf{A}}_{i,i} = \mathbf{R}_{i,i}\mathbf{A}_{i,i}\mathbf{P}_{i,i} = \mathbb{U}_i^T\mathbf{A}_{i,i}\mathbf{\Phi}_i = \mathbb{U}_i^T\mathbf{A}_{i,i}\mathbf{A}_{i,i}^{-1}\mathbb{U}_i = \mathbf{I}.$$

This fact will reduce computational and storage costs, as described in the next subsection.

To obtain a coarse-level matrix, we use a subset of the columns of $\mathbb{U}_i$ and $\mathbf{\Phi}_i$ (the first $r_H < r_h$) on each element when defining the restriction and prolongation. Since in general $\mathbf{R} \neq \mathbf{P}^T$, the coarse-level projection will be Petrov-Galerkin. Finally, note that the resulting coarse-level transformed matrix, $\widetilde{\mathbf{A}}_H$, is already contained in $\widetilde{\mathbf{A}}$, as the first r_H rows and columns of each elemental block.

3.3. p-Multigrid Implementation

This section presents details on an *in-place* implementation of the SVD-based p-multigrid method. We assume fine/coarse levels with r_h/r_H degrees of freedom per element. Instead of subscripts h and H directly on the variables, we employ a more descriptive notation: $\mathbf{i}_h$ = the set of degrees of freedom on the fine level, over all elements;

$\mathbf{i}_H$ = the set of degrees of freedom on the coarse level, over all elements; $\mathbf{i}_{h\backslash H} = \mathbf{i}_h \backslash \mathbf{i}_H$. The steps of a two-level correction cycle are then as follows:

1. Transform the system $\mathbf{A}\mathbf{u} = \mathbf{f}$ to $\widetilde{\mathbf{A}}\widetilde{\mathbf{u}} = \widetilde{\mathbf{f}}$. This transformation is done in-place by two loops over all elements;

   First loop: for each element i we

   (a) compute the SVD of the off-diagonal influence matrix, $\mathbf{A}_{i,:}\backslash\mathbf{A}_{i,i}$

   (b) overwrite $\mathbf{A}_{i,:}\backslash\mathbf{A}_{i,i}$ with the product $\mathbb{U}_i^T\,(\mathbf{A}_{i,:}\backslash\mathbf{A}_{i,i})$

   (c) overwrite $\mathbf{A}_{i,i}$ with $\mathbf{\Phi}_i$ – note, we know that the on-diagonal of $\widetilde{\mathbf{A}}$ is the identity matrix, so we do not need to store it

   (d) overwrite $\mathbf{f}_i$ with $\mathbb{U}_i^T\mathbf{f}_i$

   Second loop: for each element i we

   (a) loop over elements j adjacent to i

   (b) overwrite $\mathbf{A}_{i,j}$ with $\mathbf{A}_{i,j}\mathbf{\Phi}_j$

2. Smooth the transformed system ν_{pre} times on the fine level. Note that applying the block smoother is cheaper on the transformed system because the main diagonal block (the preconditioner) is the identity.

3. Restrict the residual by modifying the transformed source term,

$$\widetilde{\mathbf{f}}(\mathbf{i}_H) = \widetilde{\mathbf{f}}(\mathbf{i}_H) - \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_{h\backslash H})\,\widetilde{\mathbf{u}}(\mathbf{i}_{h\backslash H})$$

4. Solve the coarse-level problem,

$$\widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_H)\,\widetilde{\mathbf{u}}(\mathbf{i}_H) = \widetilde{\mathbf{f}}(\mathbf{i}_H)$$

5. The error is already prolongated from the coarse to the fine space because of the in-place storage. However, we need to undo our source modification from Step 3,

$$\widetilde{\mathbf{f}}(\mathbf{i}_H) = \widetilde{\mathbf{f}}(\mathbf{i}_H) + \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_{h\backslash H})\,\widetilde{\mathbf{u}}(\mathbf{i}_{h\backslash H})$$

6. Smooth the transformed system ν_{post} times on the fine level.

The above algorithm has no explicit application of restriction and prolongation matrices because in the transformed system these operations are trivial – the matrices consist of the identity padded with zero rows/columns. To explain Step 3 we consider the standard coarse-level error equation presented in Section 2. The fine-level residual is $\widetilde{\mathbf{r}} = \widetilde{\mathbf{f}} - \widetilde{\mathbf{A}}\,\widetilde{\mathbf{u}}$, so that the coarse-level error equation is

$$\begin{aligned}
\widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_H)\,\widetilde{\mathbf{e}}(\mathbf{i}_H) &= \widetilde{\mathbf{r}}(\mathbf{i}_H) \\
&= \widetilde{\mathbf{f}}(\mathbf{i}_H) - \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_h)\,\widetilde{\mathbf{u}}(\mathbf{i}_h) \\
&= \widetilde{\mathbf{f}}(\mathbf{i}_H) - \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_H)\,\widetilde{\mathbf{u}}(\mathbf{i}_H) - \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_{h\setminus H})\,\widetilde{\mathbf{u}}(\mathbf{i}_{h\setminus H}) \\
\Rightarrow \quad \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_H)\,\left(\widetilde{\mathbf{u}}(\mathbf{i}_H) + \widetilde{\mathbf{e}}(\mathbf{i}_H)\right) &= \widetilde{\mathbf{f}}(\mathbf{i}_H) - \widetilde{\mathbf{A}}(\mathbf{i}_H, \mathbf{i}_{h\setminus H})\,\widetilde{\mathbf{u}}(\mathbf{i}_{h\setminus H})
\end{aligned}$$

The right-hand side of this equation is precisely the modified source term after Step 3, and on the left we have the coarse-level operator multiplying the desired corrected coarse state. In summary, the proposed multigrid implementation modifies the system and right-hand side in-place using a variable transformation arising from element-wise SVD calculations; following this transformation, smoothing, restriction, and prolongation become simple operations that do not require auxiliary vectors.

4. Results

In this section we present performance comparisons of the SVD-based coarse-levels with block smoothing relative to standard p-multigrid coarse levels with both block and line smoothing. We consider a two-level cycle applied to linear advection and the linearized Euler equations in two dimensions. As a comparison metric, we look at convergence with multigrid iterations, which is not perfect because it does not account for different costs in the smoother (e.g. line more expensive than block), nor in pre-processing (e.g. the SVD calculation or block-tridiagonal factorization). As these costs can be implementation/architecture dependent we employ a simple multigrid iteration count, which does give us a rough comparison and lets us study h and p dependence of each method.

4.1. Linear Advection in Two Dimensions

The first problem we consider is two-dimensional linear advection on quadrilateral meshes of a $[0, 1]^2$ domain. The advection velocity is constant, from bottom-left to upper-right, as illustrated in Figure 7. The initial solution guess is $u = 0$, and the inflow boundary condition is $u = e^{-40(x^2+(y-0.3)^2)}$, prescribed on the bottom and left. For the baseline comparison, we use $p_h = 3 \Rightarrow r_h = 16$ on the fine level and $p_H = 2 \Rightarrow r_H = 9$ on the coarse level. The approximation space is tensor-product polynomials, and the basis is nodal Lagrange – this last point does not affect the results; a different basis would give different restriction and prolongation operators but would not change the convergence properties. Note that with the SVD approach, we are not restricted to a specific value of r_H, and we only choose r_H based on p_H to obtain a fair comparison to standard p-multigrid.

[Figure 15 about here.]

Before looking at multigrid performance, we first analyze the behavior of the singular values of the neighbor influence matrices for each element. Figure 16 presents these singular values for a sequence of three mesh refinements and one order refinement, $p_h = 4$. The plots show singular values for all elements together on one plot – these overlap due to the identical elements in the structured mesh. The vertical dashed lines on each plot mark the size of the $p_h - 1$ coarse space on each element. Note that in all cases, the number of nonzero singular values in every element is less than r_H. In fact, by considering the number of possible neighbor state variations, we can predict that the maximum number of nonzero singular values for a quadrilateral with two inflow edges will be $2p_h + 1$, and this is confirmed in Figure 16. This number will be smaller than the r_H implied by $p_H - 1$ for $p_h > 2$.

[Figure 16 about here.]

The fact that the number of singular values in each element is less than r_H means that the coarse-level solve will provide an exact solve modulo intra-element variations that will be taken care of by the block smoother. This is confirmed in Figure 17, which shows the residual convergence with multigrid iterations for the three methods tested. Note that the "block" and "line" designations refer to these smoothers with a standard $p_h - 1$ coarse-space, whereas "SVD" refers to block smoothing with the SVD coarse-space. All coarse spaces have the same size, r_H degrees of freedom per element. Standard p-multigrid exhibits p-independence but h-dependence; on the other hand, the SVD approach solves the problem exactly in one multigrid iteration. This result is insensitive to the flow angle and the solver is inherently scalable.

[Figure 17 about here.]

4.2. Linearized Euler

For a more difficult and practical problem, we turn to the linearized Euler equations. The particular problem is flow over a NACA 0012 airfoil, linearized about a solution at Mach number $M_\infty = 0.5$ and angle of attack $\alpha = 2°$, as illustrated in Figure 18. The nonlinear problem is solved on a mesh of quadratic curved quadrilaterals. The approximation space again consists of tensor-product polynomials. The source term for the linearized system is set to the residual Jacobian matrix multiplied by the exact nonlinear solution, and the initial linear state is set to $\mathbf{u} = 0$.

[Figure 18 about here.]

Figure 19 shows singular values and convergence results for the three methods for a two-level cycle between $p_h = 3 \Rightarrow r_h = 64$ and $p_H = 2 \Rightarrow r_H = 36$, on a 728-element mesh. Note, the state rank for two-dimensional Euler is 4, so that the number of degrees-of-freedom for tensor-product approximation of order p is $4(p+1)^2$ per element. In this more complicated system, the singular values vary more widely among the elements. In addition, there are elements that have more nonzero singular values than r_H, so we can

no longer expect an exact solve after one SVD-multigrid iteration. However, these extra singular values are relatively small, so we should still expect the coarse-level correction to perform well. As shown in Figure 19 this is indeed the case. The SVD approach converges in roughly three multigrid iterations, whereas standard p-multigrid with block or line smoothing exhibit much slower convergence.

[Figure 19 about here.]

To study h-dependence we next consider a uniformly-refined mesh of 2912 elements. The singular values and residual convergence histories for this case are shown in Figure 20. The distribution of singular values is nearly identical to that in Figure 19, which is expected since the uniform refinement just creates more of the same type of elements as those in the 728-element mesh. The result is that again, the SVD approach converges in just three multigrid iterations, in contrast to the traditional p-multigrid approaches which take many more iterations.

[Figure 20 about here.]

Next, we look at a higher order, $p_h = 4$ and $p_H = 3$. The singular values and residual convergence histories for this case are shown in Figure 21. We see from the singular values that now all elements have as many as or fewer than $r_H = 64$ nonzero singular values. Thus, we expect an exact solve after one multigrid iteration, and indeed this is the case, as shown in the residual convergence history.

[Figure 21 about here.]

In the results thus far, we chose r_H for the SVD approach based on the size of an order $p_H = p_h - 1$ tensor-product coarse space. However, we can choose different values of r_H. Looking at Figure 19(a), we see that the singular values decay in a step-wise fashion, with drops at $r_H = 30, 36, 48$. Figure 22 shows the results of choosing each of these values of r_H in turn in a two-level correction procedure. We see, as expected,

that the smaller the value of r_H, the worse the performance of the two-level cycle. In all cases, however, this performance is still better than that of the block or line solver.

[Figure 22 about here.]

5. Conclusions and Future Work

We have presented an element-wise algebraic approach to selecting a coarse-level approximation space for a p-type multigrid correction procedure in high-order discontinuous Galerkin discretizations of linear convection problems. This approach uses a singular-value decomposition of the neighbor-influence component of the residual Jacobian matrix for each element in order to identify the residual modes and corresponding state modes that are most "excitable" by neighbor perturbations. Preserving these modes on coarse levels keeps inter-element error transport pathways open so that convective errors can be approximated and eliminated on the coarse space, instead of by the more expensive fine-space smoother. Two-level results for advection and the linearized Euler equations confirm the algorithmic performance of this approach. Furthermore, the SVD approach uses block Jacobi as the smoother, which bodes well for fine-grained scalability.

A natural question is whether the SVD approach extends to multiple multigrid levels. In terms of implementation, the extension is straightforward: the algorithm presented in Section 3.3 applies equally well to corrections between coarse and coarser levels. But can we expect the algorithmic performance benefits to continue down to coarser levels? The answer depends on the distribution of singular values. The best scenario for the proposed approach is one in which each element has an appreciable number of zero singular values. In this case we have a "free" degree of freedom reduction to possibly very coarse levels. That is, we can reduce the number of unknowns with no effect on accuracy, since one coarse-level correction and one smoothing iteration will give us an exact solve. Next, if the singular values decay steadily or even in multiple

steps, then each coarser level will still preserve the dominant modes important for the finer level that it is correcting, and the algorithmic performance will continue to coarser levels. However, if the singular values eventually plateau for low mode numbers, the algorithmic benefits of going to coarser levels will cease. In these situations, which ideally will occur after a sizable reduction in degrees of freedom, the size of the coarse system will still likely be too large for a direct solve; one can then turn to other solvers such as Krylov methods, which now would operate on smaller systems.

Another question is about prospects for equations with diffusion. Preliminary experiments show that the singular values in such cases do not decay as quickly as in problems with only convective terms. Intuitively this is expected because in diffusion problems a state perturbation in an element will affect residuals in all of its neighbors. Therefore, the SVD-multigrid approach looses its advantage. A possible remedy in this case would be to use an operator-splitting approach to treat convective and diffusive terms differently, possibly with standard p-multigrid for the latter.

We have shown two-dimensional examples but the ideas carry over naturally to three dimensions. We expect similar benefits for convective systems at high orders. An intuitive interpretation is that the SVD approach can yield systems in which the degrees of freedom scale as $p^{\text{dim}-1}$ instead of p^{dim}, because for convection problems modes inside an element are excited only by state variations on inflow faces, and the number of independent variations scales as p to a power one lower than the dimension of the problem.

For simplicity, in our results we have assumed that r_H is the same on each element. However, we could tailor r_H to the singular value decay in each element. For some elements in which the singular values decay slowly, we could choose to keep most of the modes. Conversely, for elements that have a large number of zero singular values, we could use a much smaller r_H. In a parallel setting the partitioner would have to be aware of this non-uniformity in the coarse-level sizes for ensure load balancing, but no fundamental changes to the algorithm would be necessary.

Finally, we have considered only linear problems. For nonlinear problems, a simple extension is to apply the SVD-multigrid solver to the linearized system in Newton-Raphson. We could even use this solver as a preconditioner for a Krylov method. However, this does not alleviate the storage demands for the fine-level residual Jacobian. To address this problem, we are currently looking into extending our ideas to methods that adaptively tailor the approximation space for efficient representation of the dominant state components inside each element.

References

[1] W. Reed, T. Hill, Triangular mesh methods for the neutron transport equation, Los Alamos Scientific Laboratory Technical Report LA-UR-73-479 (1973).

[2] B. Cockburn, C.-W. Shu, Runge-Kutta discontinuous Galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (3) (2001) 173–261.

[3] B. Cockburn, B. Dong, J. Guzman, M. Restelli, R. Sacco, A hybridizable discontinuous Galerkin method for steady-state convection-diffusion-reaction problems, SIAM Journal on Scientific Computing 31 (5) (2009) 3827–3846.

[4] N. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations, Journal of Computational Physics 228 (2009) 3232–3254.

[5] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, M. Visbal, High-order CFD methods: Current status and perspective, International Journal for Numerical Methods in FluidsDOI: 10.1002/fld.3767.

[6] L. Diosady, D. Darmofal, Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations, Journal of Computational Physics 228 (2009) 3917–3935.

[7] P.-O. Persson, J. Peraire, Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations, SIAM Journal on Scientific Computing 30 (6) (2008) 2709–2733.

[8] B. Helenbrook, D. Mavriplis, H. Atkins, Analysis of p-multigrid for continuous and discontinuous finite element discretizations, AIAA Paper 2003-3989 (2003).

[9] K. J. Fidkowski, D. L. Darmofal, Development of a high-order solver for aerodynamic applications, AIAA Paper 2004-112 (2004).

[10] K. J. Fidkowski, T. A. Oliver, J. Lu, D. L. Darmofal, *p*-Multigrid solution of high–order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations, Journal of Computational Physics 207 (2005) 92–113.

[11] B. Helenbrook, H. Atkins, Application of p-multigrid to discontinuous Galerkin formulations of the Poisson equation, American Institute of Aeronautics and Astronautics Journa 44 (3) (2006) 566–575.

[12] T. Zang, Y. Wong, M. Hussaini, Spectral multigrid methods for elliptic problems, Journal of Computational Physics 48 (1982) 485–501.

[13] T. Zang, Y. Wong, M. Hussaini, Spectral multigrid methods with applications to transonic potential flow, Journal of Computational Physics 57 (1985) 43–76.

[14] E. M. Rønquist, A. T. Patera, Spectral element multigrid I. Formulation and numerical results, Journal of Scientific Computing 2(4) (1987) 389–406.

[15] Y. Maday, R. Muñoz, Spectral element multigrid II. Theoretical justification, Journal of Scientific Computing 3 (4) (1988) 323–353.

[16] R. Muñoz, Theoretical analysis of some spectral multigrid methods, Computer Methods in Applied Mechanics and Engineering 80 (1990) 287–294.

[17] S. Beuchler, Multigrid solver for the inner problem in domain decomposition methods for p-FEM, SIAM Journal on Numerical Analysis 40 (2002) 928–944.

[18] W. Heinrichs, Line relaxation for spectral multigrid methods, Journal of Computational Physics 77 (1988) 166–182.

[19] J. Lottes, P. Fischer, Hybrid multigrid/Schwarz algorithms for the spectral element method, Journal of Scientific Computing 24 (1) (2005) 45–78.

[20] M. Deville, P. Fischer, E. Mund, High-Order Methods for Incompressible Fluid Flow, Cambridge University Press, 2002.

[21] K. J. Fidkowski, A high–order discontinuous Galerkin multigrid solver for aerodynamic applications, MS thesis, M.I.T., Department of Aeronautics and Astronautics (June 2004).

[22] B. Mascarenhas, B. Helenbrook, H. Atkins, Application of the p-multigrid algorithm to discontinuous Galerkin formulations of the compressible Euler equation, AIAA Paper 2007-4331 (2007).

[23] H. Luo, J. D. Baum, R. Löhner, A *p*-multigrid discontinuous Galerkin method for the Euler equations on unstructured meshes, Journal of Computational Physics 211 (2006) 767–783.

[24] C. Nastase, D. Mavriplis, High-order discontinuous Galerkin methods using an hp-multigrid approach, Journal of Computational Physics 213 (2006) 330–357.

[25] T. A. Oliver, K. J. Fidkowski, D. L. Darmofal, Multigrid solution for high–order discontinuous Galerkin discretization of the compressible Navier-Stokes equations, in: Third International Conference on Computational Fluid Dynamics, Toronto, Canada, 2004.

[26] T. A. Oliver, Multigrid solution for high–order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations, MS thesis, M.I.T., Department of Aeronautics and Astronautics (August 2004).

[27] K. Shahbazi, D. Mavriplis, N. Burgess, Multigrid algorithms for high-order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations, Journal of Computational Physics 228 (2009) 7917–7940.

[28] H. Atkins, B. Helenbrook, Numerical evaluation of p-multigrid method for the solution of discontinuous Galerkin discretizations of diffusive equations, AIAA Paper 2005-5110 (2005).

[29] A. Brandt, Guide to Multigrid Development, Springer-Verlag, 1982.

List of Figures

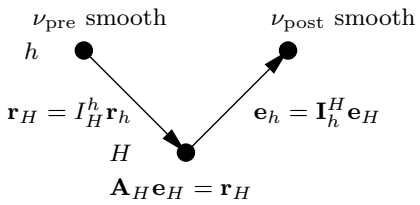
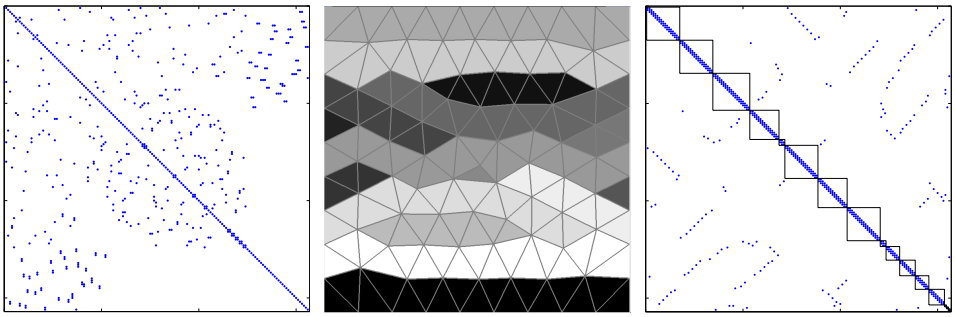
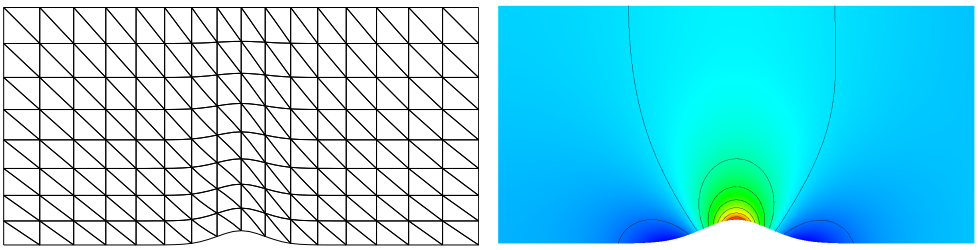


Figure 1: Illustration of a two-level correction cycle, where the fine (h) and coarse (H) levels are denoted by subscripts.



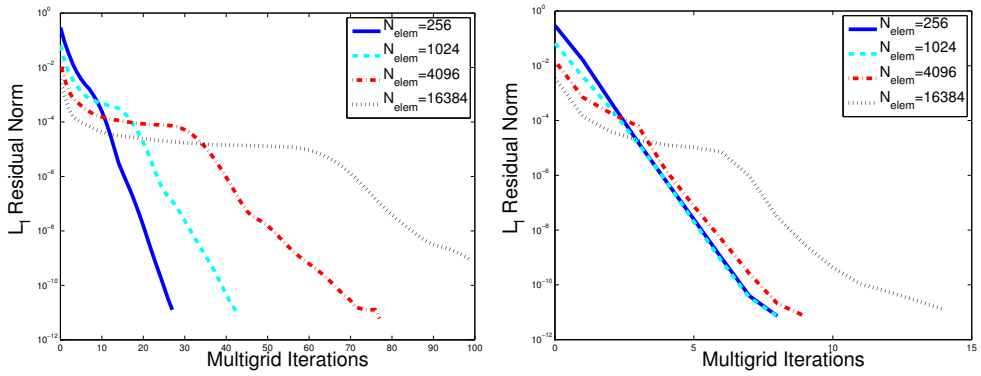
(a) Jacobian matrix, **A** (b) Mesh and lines (c) **A**, elements reordered

Figure 2: An element-block smoother inverts only the block diagonal of the residual Jacobian matrix, **A**, whereas an element-line Jacobi smoother solves block tridiagonal systems formed by ordering elements along strongest coupling. Block-ILU and Gauss Seidel smoothers share characteristics with the element-line smoother.



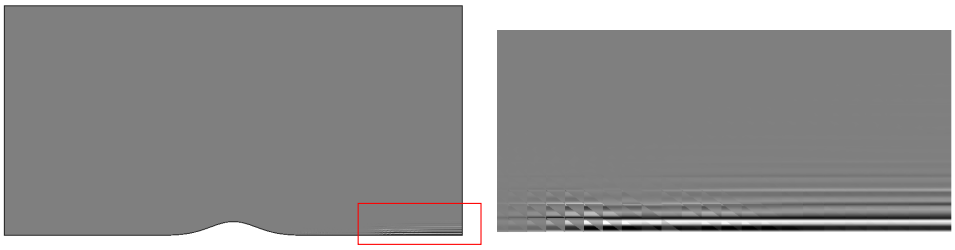
(a) A coarse mesh (b) Mach number contours

Figure 3: Inviscid flow over a bump: the elements are curved using a cubic mapping from reference to global space.



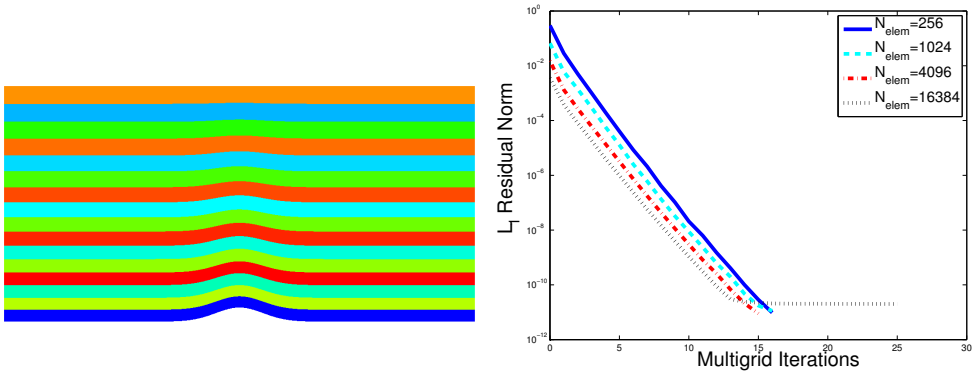
(a) $\nu_{\text{pre}} = \nu_{\text{post}} = 2$ (b) $\nu_{\text{pre}} = \nu_{\text{post}} = 20$

Figure 4: Inviscid flow over a bump: mesh dependence of the p-multigrid V-cycle with element Jacobi smoothing as a function of the number of pre-correction and post-correction smoothing iterations. Two levels were used, $p_h = 3$ and $p_H = 2$, with an exact solve on $p_H = 2$.



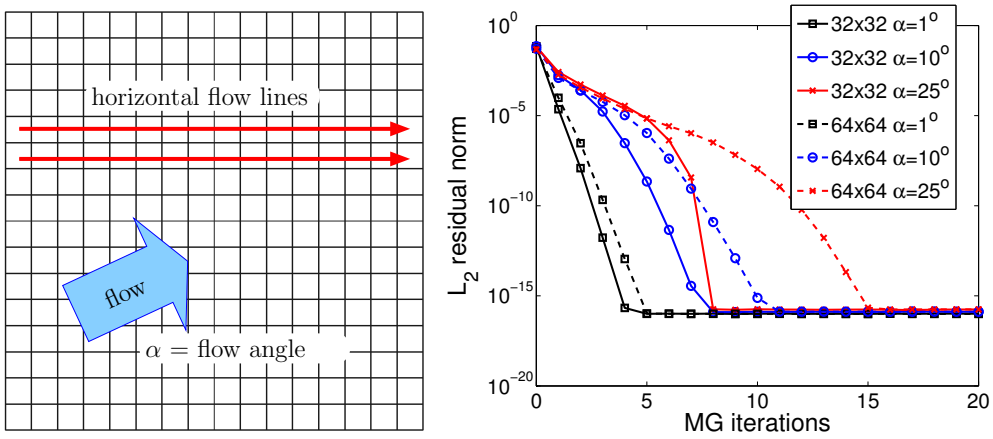
(a) Far view of density error (b) Near view

Figure 5: Inviscid flow over a bump: slowest converging mode corresponding to the finest grid in Figure 4a. The density error is shown at ten multigrid iterations, which is when the residual convergence begins to plateau. The error is high frequency orthogonal to the streamlines, and low frequency in the streamwise direction.



(a) Lines of elements (b) Residual convergence

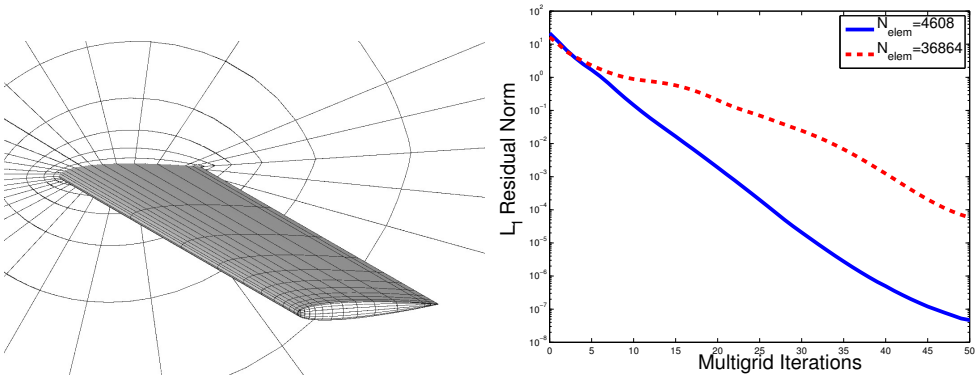
Figure 6: Inviscid flow over a bump: mesh independence of the p-multigrid V-cycle with element-line Jacobi smoothing, for $\nu_{\text{pre}} = \nu_{\text{post}} = 2$. Two levels were used, $p = 3$ and $p = 2$, with an exact solve on $p = 2$.



(a) Line/flow alignment test

(b) Residual convergence

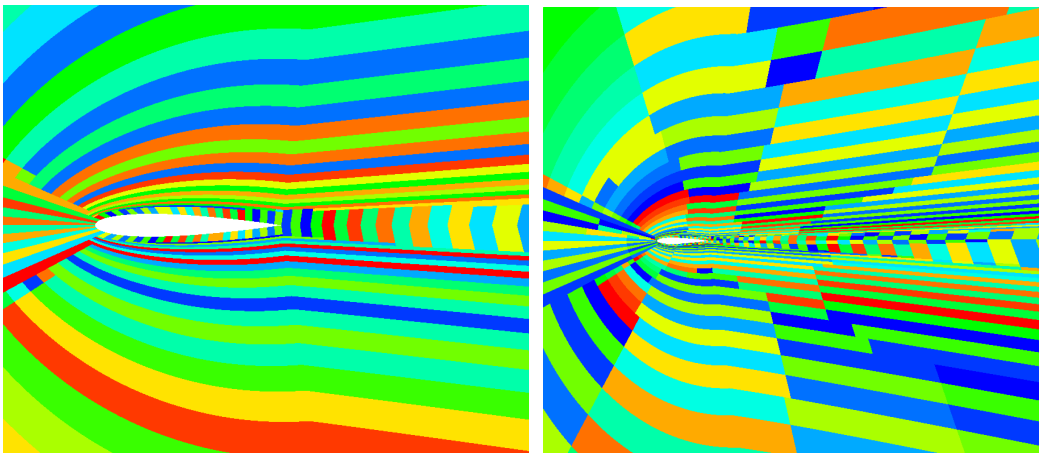
Figure 7: Two-dimensional advection with horizontal lines: flow/line alignment test for two-level p-multigrid with the element-line smoothing. Two levels were used, $p_h = 3$ and $p_H = 2$, with an exact solve on $p_H = 2$, and $\nu_{\text{pre}} = \nu_{\text{post}} = 2$.



(a) Hexahedral mesh of a wing

(b) Residual convergence

Figure 8: Inviscid flow over a wing: h-dependence of p-multigrid with the element-line smoother. Lines were created using a physics-based coupling measure computed from the inviscid flux [10], and due to the structure of the mesh they align reasonably well with the flow in most of the domain. Two levels were used, $p_h = 2$ and $p_H = 1$, with an exact solve on $p_H = 1$, and $\nu_{\text{pre}} = \nu_{\text{post}} = 2$.



(a) 1 core (b) 32 cores

Figure 9: Laminar Navier-Stokes flow over an airfoil: parallel degradation of the element line smoother: lines are cut by partition boundaries.

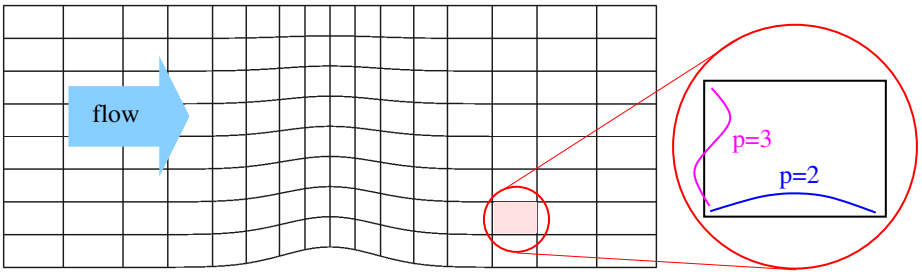
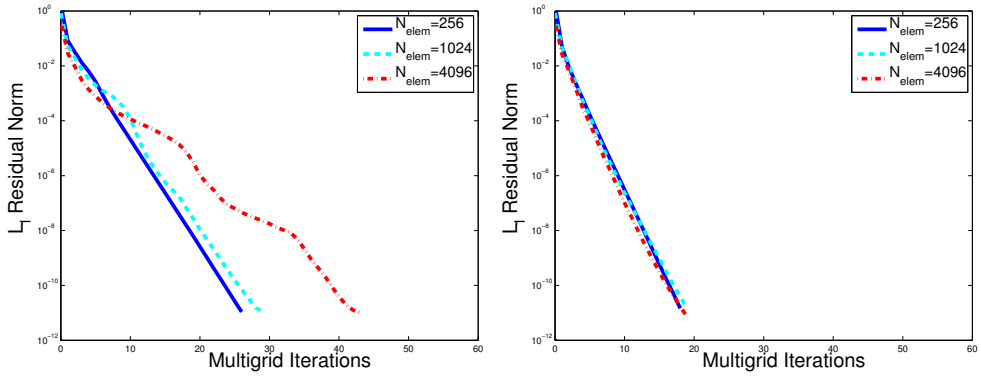
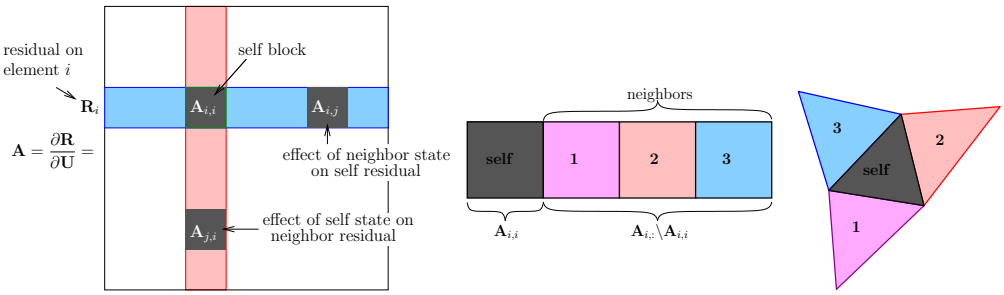


Figure 10: Inviscid flow over a bump: construction of a semi-coarsened approximation space on each element, in this case by taking advantage of the mesh structure and alignment with the flow.



(a) Coarse space: $(p=2)\otimes(p=2)$ (b) Coarse space: $(p=2)\otimes(p=3)$

Figure 11: Inviscid flow over a bump: convergence of p-multigrid with full- and semi-order coarsening, using a tensor-product Lagrange basis. A two-grid cycle is used with $\nu_{\text{pre}} = \nu_{\text{post}} = 5$. Note the h-independence of the residual convergence when using semi-coarsening of the tensor-product approximation space.



(a) Jacobian matrix, $\mathbf{A}$ (b) Compressed block row for element i

Figure 12: Schematic of the residual Jacobian matrix and its compressed block row storage.

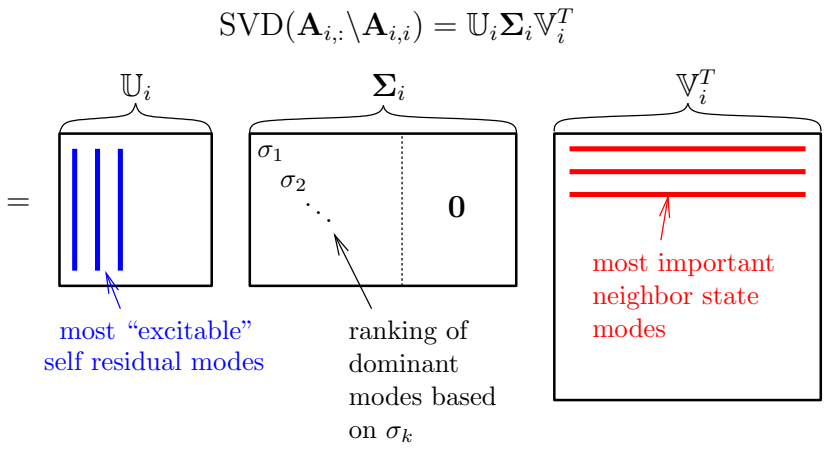
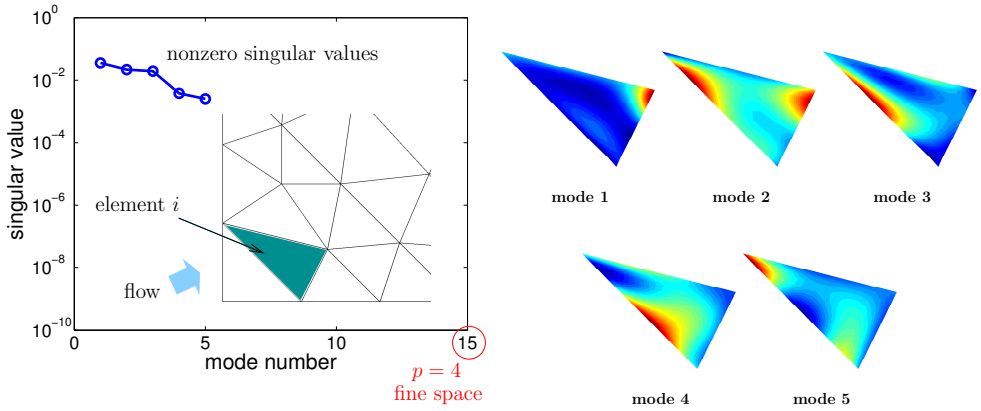


Figure 13: Singular-value decomposition of the neighbor influence matrix, $\mathbf{A}_{i,:}$.



(a) Singular values (b) Excitable modes

Figure 14: Demonstration of neighbor-influence singular value behavior for a linear advection problem on an unstructured mesh. Singular values and the corresponding "excitable modes" are shown for the highlighted element.

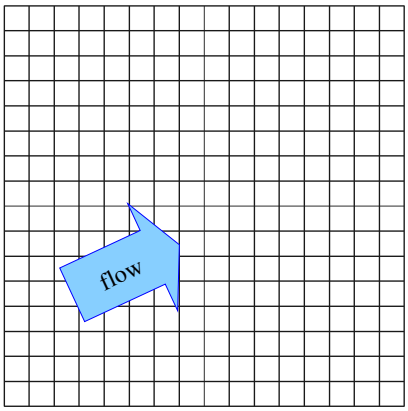
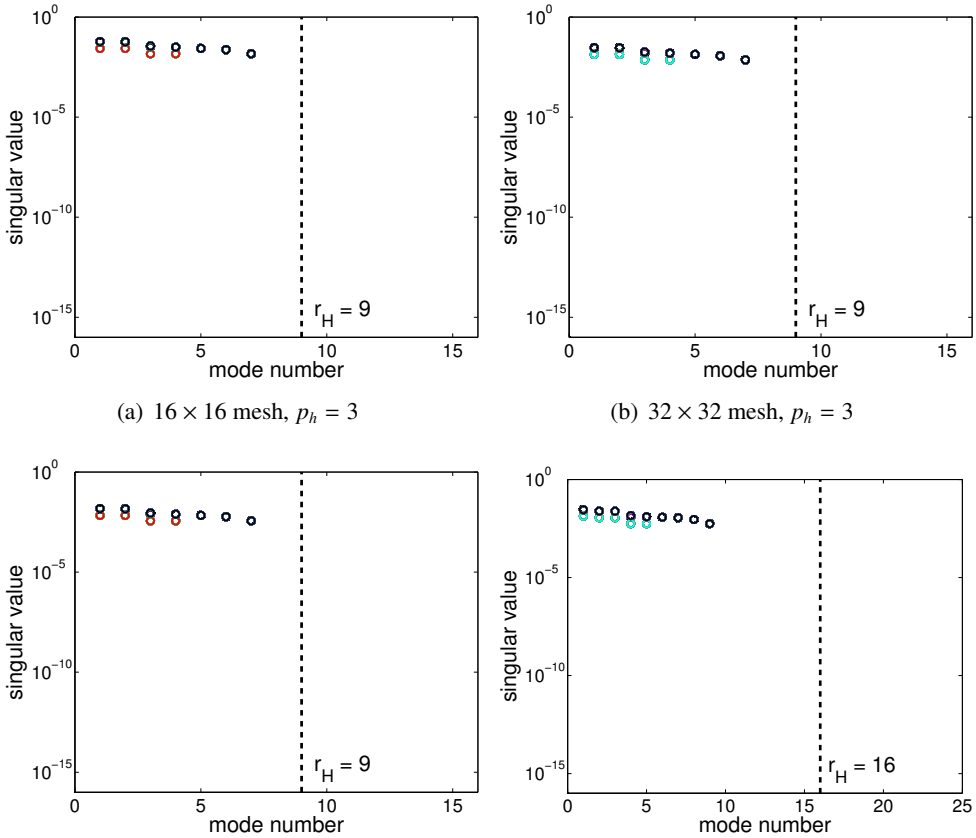


Figure 15: 2D Linear advection on a quadrilateral mesh, $\alpha = 25^o$.



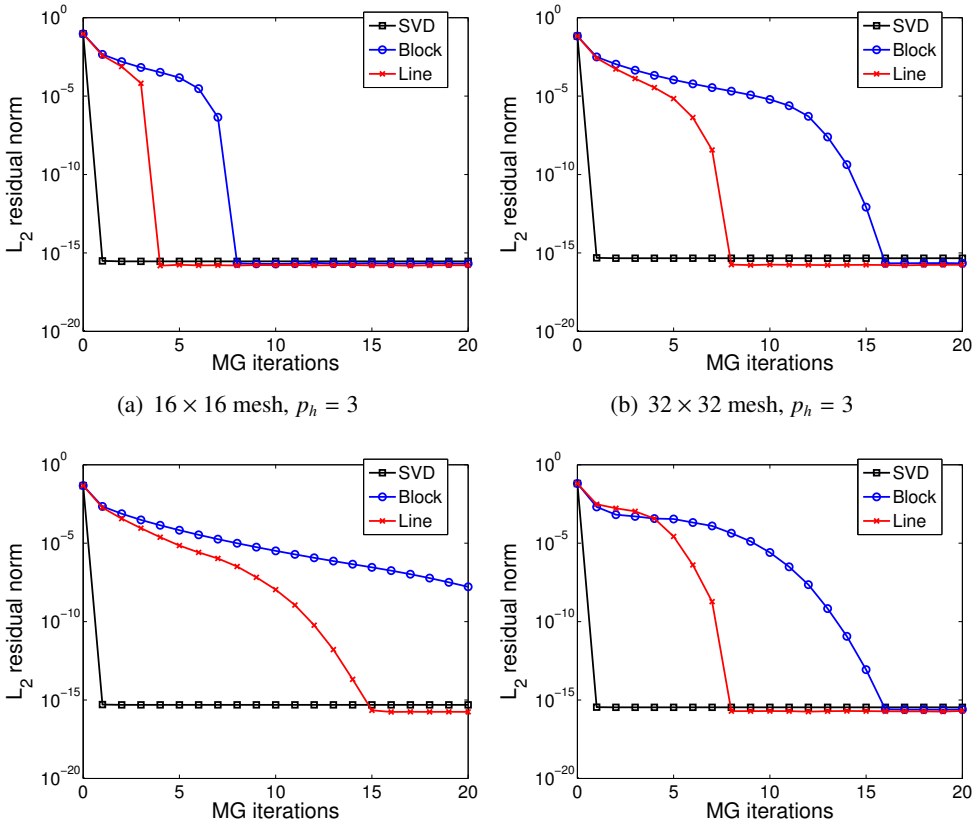
(a) 16×16 mesh, $p_h = 3$

(b) 32×32 mesh, $p_h = 3$

(c) 64×64 mesh, $p_h = 3$

(d) 32×32 mesh, $p_h = 4$

Figure 16: Comparison of singular values for an advection problem on a structured mesh. Singular values of all elements are shown together on each plot, with colors picked randomly to distinguish between singular values.



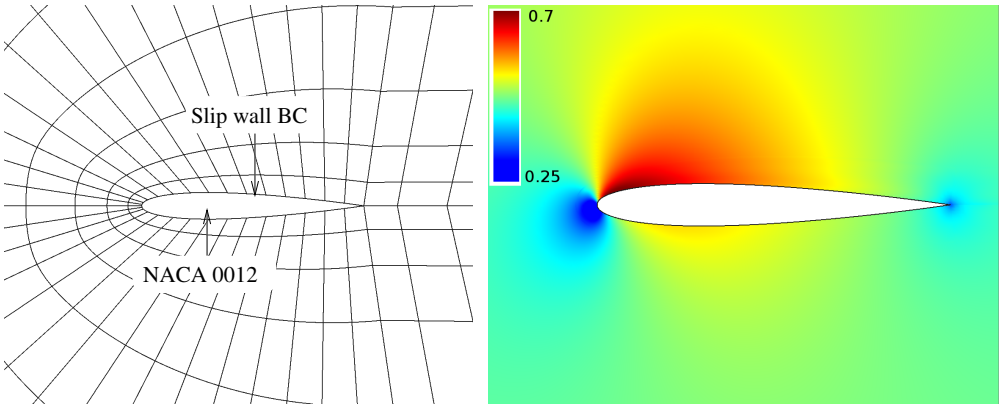
(a) 16×16 mesh, $p_h = 3$

(b) 32×32 mesh, $p_h = 3$

(c) 64×64 mesh, $p_h = 3$

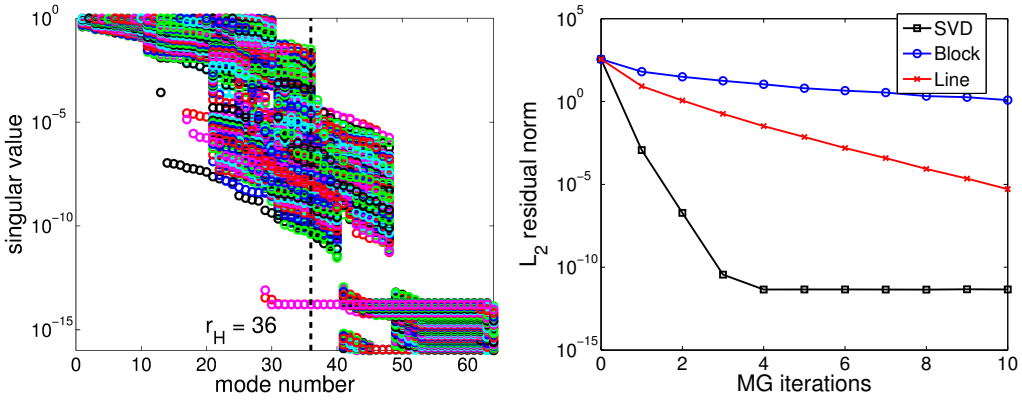
(d) 32×32 mesh, $p_h = 4$

Figure 17: Comparison of two-level correction convergence rates for advection on a structured mesh.



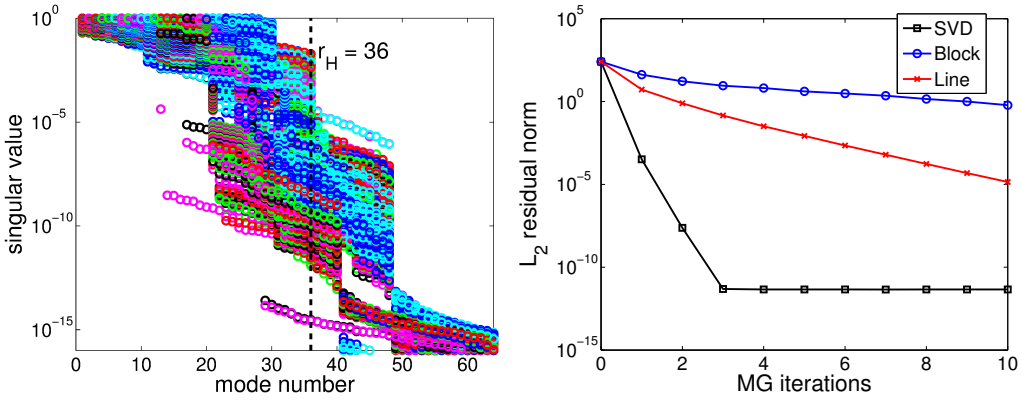
(a) Problem setup (b) Mach contours

Figure 18: Problem setup ($\alpha = 2°, M_\infty = 0.5$) and Mach number contours for the steady-state flow solution over a NACA 0012 airfoil. This is the state about which we linearize for the linearized Euler solver.



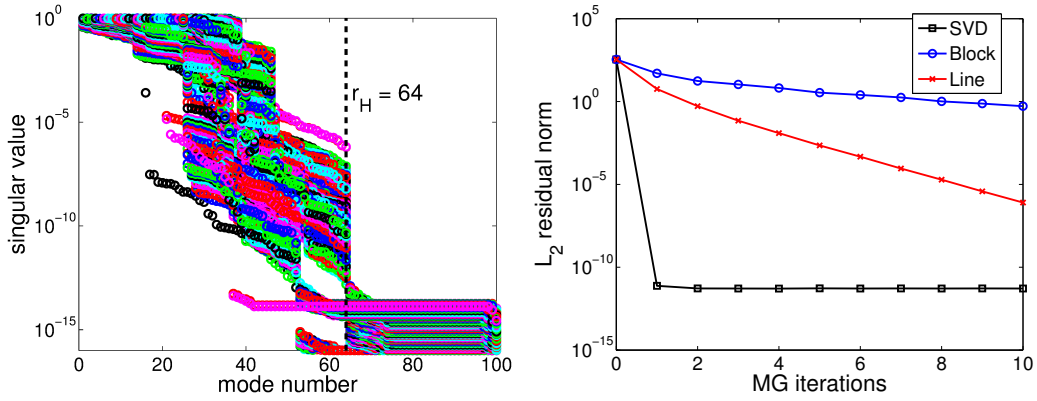
(a) Singular values (b) Convergence

Figure 19: Linearized Euler flow over a NACA 0012 airfoil: singular values and residual convergence of a two-grid correction cycle for a 728-element mesh, with $p_h = 3$ and $p_H = 2$. Tensor product basis functions are used.



(a) Singular values (b) Convergence

Figure 20: Linearized Euler flow over a NACA 0012 airfoil: singular values and residual convergence of a two-grid correction cycle for a fine, 2912-element mesh, using $p_h = 3$ and $p_H = 2$.



(a) Singular values (b) Convergence

Figure 21: Linearized Euler flow over a NACA 0012 airfoil: singular values and residual convergence of a two-grid correction cycle for a 728-element mesh, using $p_h = 4$, $p_H = 3$.

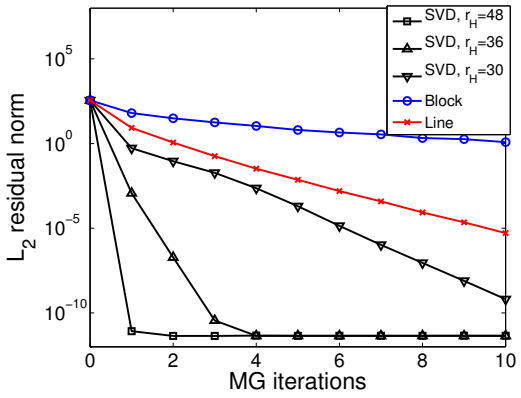


Figure 22: Linearized Euler flow over a NACA 0012 airfoil: study of the effect of the coarse-space size, r_H, for a 728-element mesh with $p_h = 3 \Rightarrow r_h = 64$.